

# **DEVELOPING DATA ENTRY APPLICATIONS WITH SAS/FSP**

Statistical Support Section  
Office of Computing Resources and Services  
Center for Information Technology  
National Institutes of Health

March 17, 1999

## **Course Description**

The procedure FSEDIT in SAS/FSP allows the user to design and use custom data entry screens to create, edit and browse SAS data sets. It provides tools to set attributes for each data entry field. A Screen Control Language (SCL) program can be used with FSEDIT to check for errors and to display computed values in special fields when the data is being entered. It can also be used to assign user-defined help windows and pull-down menus with the data entry form.

## **Course Objectives**

After completing this course you should be able to

- create SAS data sets for data entry
- create custom data entry screens
- set attributes for data entry fields
- use SCL to check for errors and display computed values
- add and modify observations
- locate observations meeting certain criteria
- create a help window
- associate customized function keys with the application
- create customized pull-down menus

## **Environments and Documentation**

SAS/FSP is available under the MVS environment on the NIH IBM mainframe, the NIH ALW network of UNIX workstations, and on Windows, OS/2 and Mac environments.

Documentation for SAS/FSP is available from the CIT Technical Information Office (301-594-3278) or by using the CIT Web magazine LiveWire ([livewire.nih.gov](http://livewire.nih.gov)). These are the guides available:

- SAS/FSP Software: Usage and Reference, Version 6, First Edition
- SAS Screen Control Language, Reference, Version 6, First Edition
- SAS Screen Control Language, Usage, Version 6, First Edition

## **Conventions Used in These Notes**

For mainframe users and for those who prefer to use SAS commands instead of SAS menus the appropriate commands will be displayed in parenthesis and after the characters "=>", e.g. (=>SUBMIT) where necessary.

## Description of Sample Application

In this lecture we will be creating an FSEDIT application to enter patient data into a SAS data set using a customized screen.

Before you start to build the application it helps to write down a general outline of what you want the application to include or do, such as, variable names, variable attributes, variables to compute, data errors to check for and so on. The following is a general outline of the application we will create.

### Data Set Variables

| NAME     | TYPE | LEN | LABEL          | FORMAT    | INFORMAT  |
|----------|------|-----|----------------|-----------|-----------|
| subno    | c    | 4   | Subject Number |           |           |
| name     | c    | 30  | Subject Name   |           |           |
| bday     | n    |     | Birthdate      | mmddyy10. | mmddyy10. |
| sex      | c    | 1   |                |           |           |
| age      | n    |     |                |           |           |
| pregnant | c    | 1   |                | \$preg.   |           |
| drug     | c    | 1   |                |           |           |
| resp1    | n    |     |                |           |           |
| resp2    | n    |     |                |           |           |
| resp3    | n    |     |                |           |           |

### Data Set Variable Attributes

|          |                                      |
|----------|--------------------------------------|
| sex      | define format: F=FEMALE M=MALE       |
| age      | minimum 21                           |
| pregnant | define format: N=NO Y=YES            |
| drug     | define format: A=ASPIRIN I=IBUPROFEN |

## In the SCL Program

Compute the screen variable:

```
avgresp=mean(resp1,resp2,resp3);
```

Check for these errors:

```
if drug not in('A','I') then write error message;  
if (sex='M' and pregnant='Y') then write error message;
```

## Building the Data Entry Application

We will follow these steps to build the application:

1. create the SAS library with a LIBNAME statement
2. create a permanent format library using PROC FORMAT
3. create a data entry screen using PROC FSEDIT
4. write an SCL program that checks for errors
5. set function keys for the application
6. create a help window using the BUILD window of SAS/AF
7. create pull-down menus using the Base SAS procedure PMENU

## Creating the SAS Library

To create the SAS library where we will store all the files of our application we SUBMIT the LIBNAME statement shown below. (= > SUBMIT) The directory must already exist.

```
libname library 'c:\testdrug';
```

## Saving Formats Permanently

To associate user-defined formats with the data permanently you must first create a permanent format library. This is done by using PROC FORMAT with the LIBRARY= option. After the equal sign specify a valid libref. To save the format library in the same SAS library as the data set specify the same libref as in the LIBNAME statement you entered (e.g. LIBRARY).

**Note:** If you will be using formats from a format library you created it is convenient to use the libref LIBRARY since SAS searches the libraries associated with the libref LIBRARY first when you reference a user-defined format.

The program below creates a format library with three formats: \$SEX, \$PREG and \$DRUG.

```
proc format library=library;
  value $sex   'F'='FEMALE'
               'M'='MALE' ;

  value $preg  'N'='NO'
               'Y'='YES' ;

  value $drug  'A'='ASPIRIN'
               'I'='IBUPROFEN' ;
run;
```

On Windows, SAS assigns the name FORMATS.SC2 to the format library. On MVS, it is called FORMATS.

## Invoking PROC FSEDIT

Start the FSEDIT procedure by submitting statements like these:

```
proc fseedit new=library.testdrug
             screen=library.testdrug.form1
             label;
run;
```

The option NEW= specifies the two-part name of the SAS data set to create. If the data set exists use DATA= instead of NEW=. The first part is the libref and the second part is the SAS data set name.

The option SCREEN= specifies the name of the SAS catalog and the name of the screen to create. The first part of the name is the libref, the second part is

the SAS catalog name and the third part is the screen name. The third part can be omitted and defaults to FSEDIT. SAS adds a fourth part to the screen name: SCREEN.

Notice that the name of the data set and the catalog are the same, i.e. TESTDRUG. The names don't have to be the same. On Windows, SAS data set names have the extension .SD2 and SAS catalog names have the extension .SC2.

The word LABEL is optional and indicates that the labels of the variables, if any, should be used in the new screen instead of the variable names.

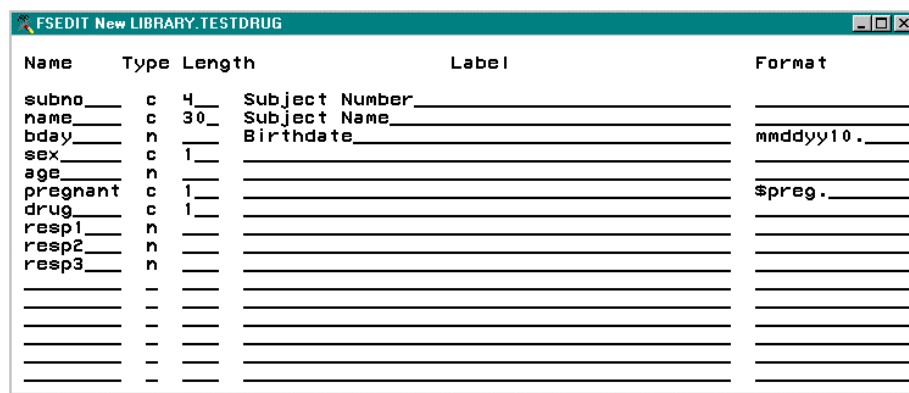
In the catalog LIBRARY.TESTDRUG we will save:

- a data entry form called FORM1
- a set of function keys called SETKEYS
- a set of pull-down menus called MENUS, and
- a help window called MYHELP

## Defining Variables

After submitting the FSEDIT program the FSEDIT New window is displayed.

In this window you define variable names, variable types (C or N), variable lengths, variable labels, and variable formats and informats. To define informats select Format/Informat from the Locals menu. (= > RIGHT)

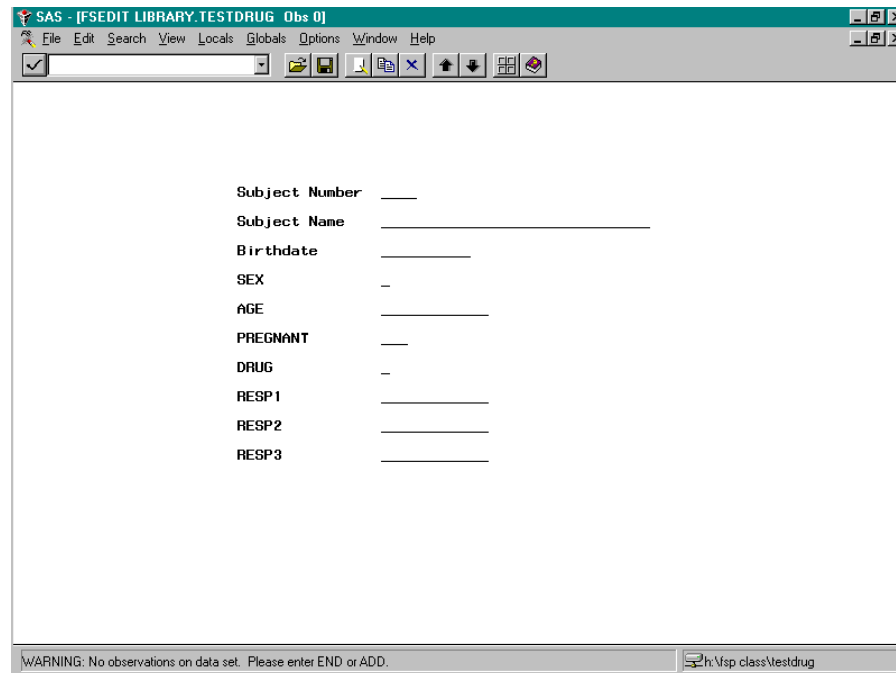


| Name     | Type | Length | Label          | Format    |
|----------|------|--------|----------------|-----------|
| subno__  | c    | 4__    | Subject Number |           |
| name__   | c    | 30__   | Subject Name   |           |
| bday__   | n    |        | Birthdate      | mmddyy10. |
| sex__    | c    | 1__    |                |           |
| age__    | n    |        |                |           |
| pregnant | c    | 1__    |                | \$preg.   |
| drug__   | c    | 1__    |                |           |
| resp1__  | n    |        |                |           |
| resp2__  | n    |        |                |           |
| resp3__  | n    |        |                |           |
|          | -    |        |                |           |
|          | -    |        |                |           |
|          | -    |        |                |           |
|          | -    |        |                |           |
|          | -    |        |                |           |

After entering this information select End from the File menu. (= > END)

## Customizing the Screen and Identifying the Variables

After you exit the FSEDIT New window a window will be displayed that contains the default data entry screen. You can modify this form so it is organized as you wish. This is the default screen:



The screenshot shows the SAS FSEDIT window titled "SAS - [FSEDIT LIBRARY.TESTDRUG Obs 0]". The menu bar includes File, Edit, Search, View, Locals, Globals, Options, Window, and Help. Below the menu bar is a toolbar with various icons. The main area displays a data entry form with the following fields:

|                |       |
|----------------|-------|
| Subject Number | _____ |
| Subject Name   | _____ |
| Birthdate      | _____ |
| SEX            | -     |
| AGE            | _____ |
| PREGNANT       | _____ |
| DRUG           | -     |
| RESP1          | _____ |
| RESP2          | _____ |
| RESP3          | _____ |

At the bottom of the window, a status bar displays the message: "WARNING: No observations on data set. Please enter END or ADD." and the file path "2h\vsp class\testdrug".

To modify the screen select Modify screen from the Locals menu. (= > MODIFY) It will prompt you for a password. You can assign a password to your application or leave it blank if you don't need one. Click OK when done even if you don't specify a password.

The FSEDIT menu is now displayed. These are the options:

- 1 Information about screen modification
- 2 Screen Modification and Field Identification
- 3 Edit Program Statements and Compile
- 4 Assign Special Attributes to Fields
- 5 Modification of General Parameters
- 6 Browse Program Statements

Select 2 to modify the screen. (= > 2)

Now you can design the screen the way you want it to look. If you have saved your design in a text file you can include the file by selecting Open from the File menu then selecting Read file. (= > INC '*filename*') In the screen you must



use underscores for the variable fields. Each field must have at least one blank before and after the underscores.

If you need to display numbers in this editor window select Options from the Edit menu then select Numbers. (= > NUMS)

This is what the screen looks like after modifying it:

Subject Number: \_\_\_\_

Subject Name: \_\_\_\_\_

Birthdate: \_\_\_\_ Sex: \_ Age: \_\_\_\_  
mm/dd/yyyy

Pregnant: \_\_\_\_ Drug: \_

Response 1: \_\_\_\_ Response 2: \_\_\_\_ Response 3: \_\_\_\_

Average Response: \_\_\_\_

Goback Help

NOTE: NUMS is set to OFF. h:\sp class\vestdrug

Select End from the File menu to exit this window and save the screen.  
(= > END)

You will be prompted with the question: "Did you create any computational or repeated fields (Y or N)?" If you created any fields that will be calculated in the SCL program (e.g. the average response variable) then specify Y. The FSEDIT Names window is displayed so you can enter the variable attributes:

| Name   | Type | Format | Informat |
|--------|------|--------|----------|
| AVGRES | N    | 5.2    |          |
|        |      |        |          |
|        |      |        |          |
|        |      |        |          |
|        |      |        |          |
|        |      |        |          |
|        |      |        |          |
|        |      |        |          |
|        |      |        |          |
|        |      |        |          |

Select End from the File menu when done. (= > END)

Now SAS displays the FSEDIT Identify window. One by one, SAS will ask you to place the cursor on the field that identifies each variable. You have the option of selecting Unwanted from the Locals menu if you don't want a variable to be displayed. (= > UNWANTED) For example, maybe you don't have the data for a variable now but will receive it later.

SAS

File View Locals Globals Options Window Help

FSEDIT Identify Screen 1

Subject Number:

Subject Name:

Birthdate:  mm/dd/yyyy Sex:  Age:

Pregnant:  Drug:

Response 1:  Response 2:  Response 3:

Average Response:

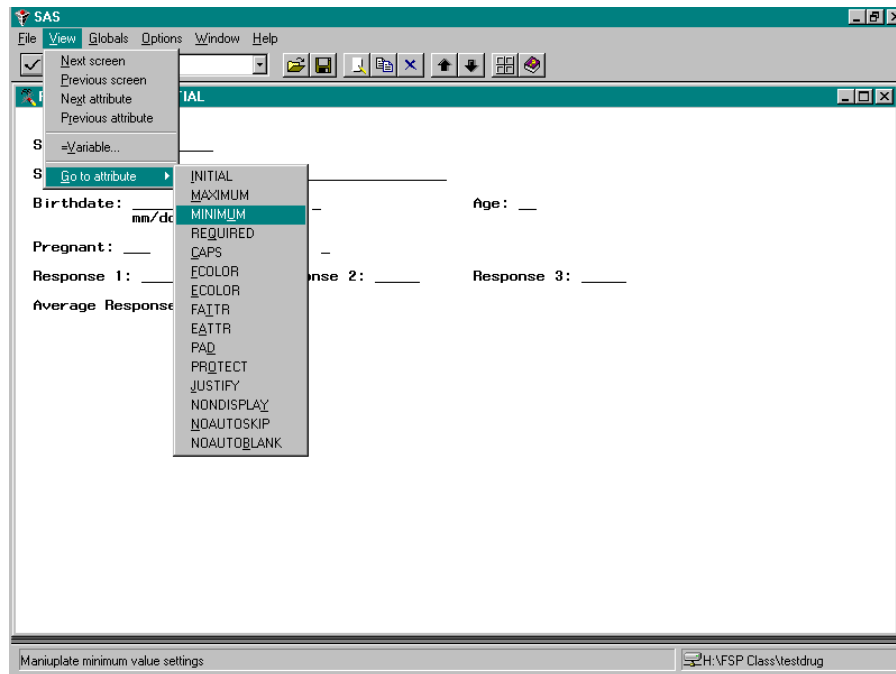
Please put cursor on field: SUBNO and press ENTER ... or UNWANTED

H:\FSP Class\vestdrug

After all the fields are identified select End from the File menu. The FSEDIT menu will be displayed again. (= > END)

## Assigning Special Attributes to Variables

Select 4 to assign a minimum value of 21 to AGE. ( $\Rightarrow$ 4) Then select Go to Attribute from the View menu and select Minimum from the list of attributes. ( $\Rightarrow$ MINIMUM) Move the cursor to AGE and type 21.



Select End from the File menu to exit this window. The FSEDIT menu will be displayed. ( $\Rightarrow$ END)

## Invoking the SAS Data Set with FSEDIT

If you are not creating an SCL program you can exit the FSEDIT session at this point or start entering data.

To start entering data select Add new record from the Edit menu.

If you exit FSEDIT and enter your data later on, invoke FSEDIT with the following statements:

```
proc fseedit data=library.testdrug
              screen=library.testdrug.form1;
run;
```

## Editing the Screen Control Language Program

Select 3 to create an SCL program. (= > 3)

To display line numbers select Options from the Edit menu then select Numbers. (= > NUMS) If you have saved the program in a text file you can include the file by selecting Open from the File menu then selecting Read file. (INC '*filename*')

An SCL program can consist of the following sections:

|         |  |
|---------|--|
| FSEINIT | initializes the application before any observations are displayed            |
| INIT    | runs after reading an observation and before displaying it                   |
| MAIN    | runs after the observation is displayed, it is modified and Enter is pressed |
| TERM    | runs before the next observation is displayed                                |
| FSETERM | runs when the FSEDIT session ends with the END command                       |

You can also include variable-labeled sections, where the label is the name of a variable in the screen. These sections run before the MAIN section and only if the user modifies that variable and presses the Enter key.

Each section in the SCL program starts with the section name followed by a colon, and ends with a RETURN statement that transfers control back to FSEDIT. The FSEINIT and FSETERM sections are not required.

Here is the SCL program for this application:

```
FSEINIT:
  control error label; ❶
return;

INIT:
  avgresp=mean(resp1,resp2,resp3); ❷
return;

DRUG:
  if drug not in('A','I')then do; ❸
    erroron drug;
    _msg_='Invalid data for DRUG';
  end;
return;

MAIN:
  if (sex='M' and pregnant='Y') then do; ❹
    erroron pregnant sex;
    _msg_='Invalid data for SEX or PREGNANT';
    return;
  end;
  else erroroff pregnant sex;
  avgresp=mean(resp1,resp2,resp3); ❺
return;

TERM:
return;
```

- ❶ By default, SAS runs the MAIN section only after the user has corrected the values for all the fields that have an error flag set to on. The ERROR options of the CONTROL statement tells SAS that it should instead run the MAIN section whenever the user presses Enter even if some fields have an error flag set to on. See ❸ below. The option LABEL specifies that we are using a labeled section (i.e. DRUG).
- ❷ Before an observation is displayed the value of the average response is calculated based on the current values of RESP1, RESP2 and RESP3.
- ❸ Here we check that the user entered a valid value for DRUG.
- ❹ Here we cross-validate the values of SEX and PREGNANT. If the user enters yes for a male the ERRORON statement sets error flags for PREGNANT and SEX. We also set the special variable \_MSG\_ to a string containing an error message to be displayed at the bottom of the screen. The RETURN statement transfers execution to PROC FSEDIT. Without the

CONTROL ERROR statement above, the user would have to enter data in both PREGNANT and SEX for SAS to set the error flags to off even if a value was correct. If the person made a mistake in only one of these fields we would like SAS to run the MAIN section again as soon as the person changes that value and presses Enter even if the other value was not changed (i.e. it's error flag is set to on).

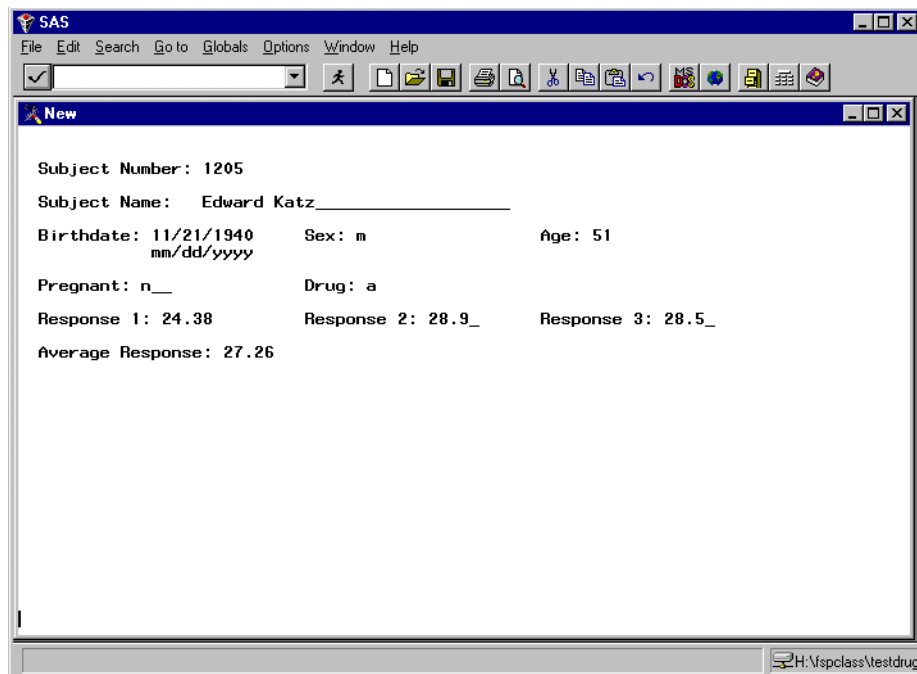
- ⑤ Calculates AVGRES with the current (maybe modified) values of RESP1, RESP2 and RESP3.

Selecting End from the File menu compiles the program and returns to the FSEDIT menu. (= > END) If there are any errors the error messages will appear in the LOG window. Press GOBACK. (= > END)

Now you can start entering the data by selecting Add new record from the Edit menu or selecting End from the File menu to end PROC FSEDIT. (= > ADD) (= > END)

## Entering Data

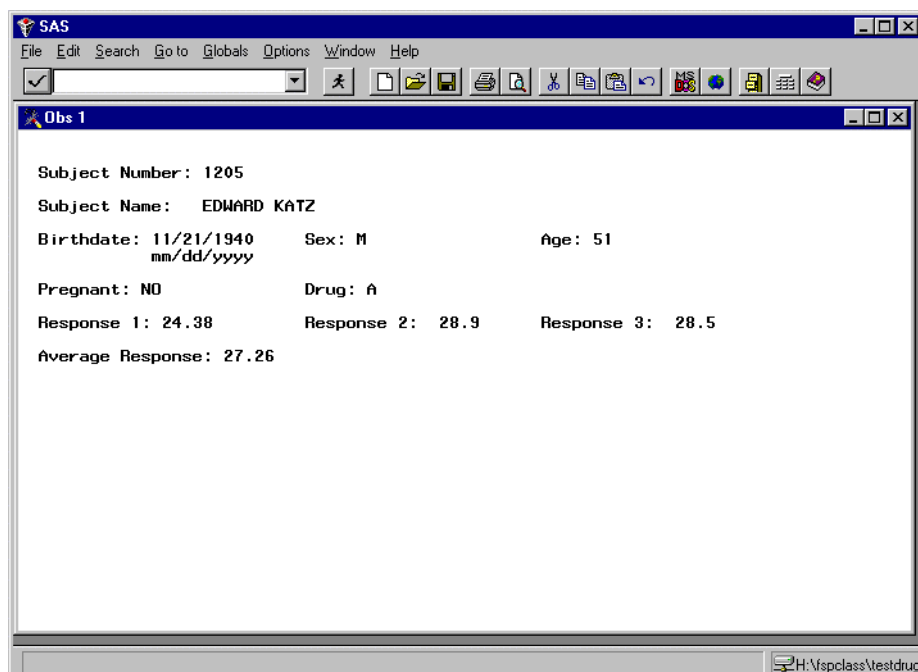
To add data select Add new record from the Edit menu then enter data in the fields using the Tab keys or mouse to move from one field to another.  
(=>ADD) Press Enter when you finish entering all the data for a screen.



The screenshot shows the SAS 'New' window with the following data entry fields:

|                                     |                  |                  |
|-------------------------------------|------------------|------------------|
| Subject Number: 1205                |                  |                  |
| Subject Name: Edward Katz           |                  |                  |
| Birthdate: 11/21/1940<br>mm/dd/yyyy | Sex: m           | Age: 51          |
| Pregnant: n                         | Drug: a          |                  |
| Response 1: 24.38                   | Response 2: 28.9 | Response 3: 28.5 |
| Average Response: 27.26             |                  |                  |

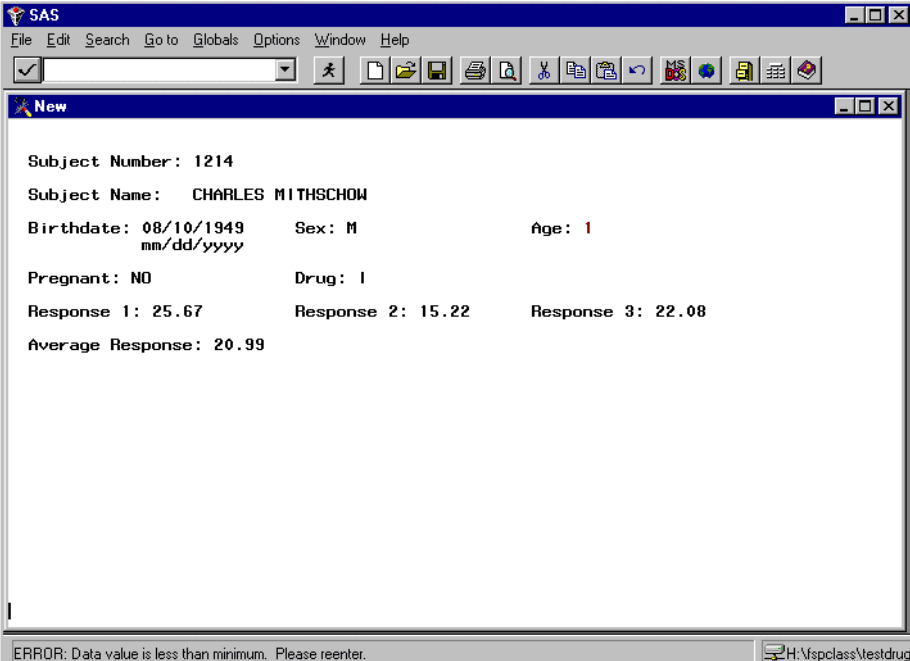
After you press the Enter key FSEDIT displays the formatted values, checks that the special attributes of the fields are valid then processes the SCL program. This is what the screen looks like after pressing Enter:



The screenshot shows the SAS 'Obs 1' window with the following formatted data:

|                                     |                  |                  |
|-------------------------------------|------------------|------------------|
| Subject Number: 1205                |                  |                  |
| Subject Name: EDWARD KATZ           |                  |                  |
| Birthdate: 11/21/1940<br>mm/dd/yyyy | Sex: M           | Age: 51          |
| Pregnant: NO                        | Drug: A          |                  |
| Response 1: 24.38                   | Response 2: 28.9 | Response 3: 28.5 |
| Average Response: 27.26             |                  |                  |

What follows are other examples of entering data, including data that is entered incorrectly.



SAS

File Edit Search Go to Globals Options Window Help

New

Subject Number: 1214

Subject Name: CHARLES MITHSCHOW

Birthdate: 08/10/1949      Sex: M      Age: 1  
mm/dd/yyyy

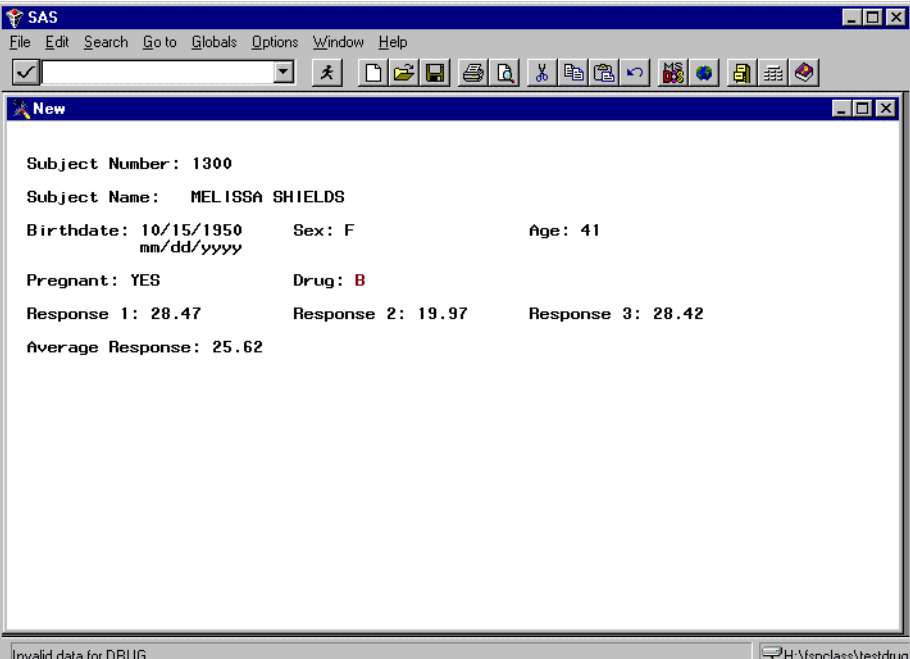
Pregnant: NO      Drug: I

Response 1: 25.67      Response 2: 15.22      Response 3: 22.08

Average Response: 20.99

ERROR: Data value is less than minimum. Please reenter.

H:\mspclass\testdrug



SAS

File Edit Search Go to Globals Options Window Help

New

Subject Number: 1300

Subject Name: MELISSA SHIELDS

Birthdate: 10/15/1950      Sex: F      Age: 41  
mm/dd/yyyy

Pregnant: YES      Drug: B

Response 1: 28.47      Response 2: 19.97      Response 3: 28.42

Average Response: 25.62

Invalid data for DRUG

H:\mspclass\testdrug



SAS

File Edit Search Go to Globals Options Window Help

✓

New

Subject Number: 1210

Subject Name: ROBERT MILLER

Birthdate: 02/15/1963      Sex: M      Age: 29  
mm/dd/yyyy

Pregnant: YES      Drug: A

Response 1: 34.5      Response 2: 30.25      Response 3: 32.23

Average Response: \_\_\_\_\_

Invalid data for SEX or PREGNANT

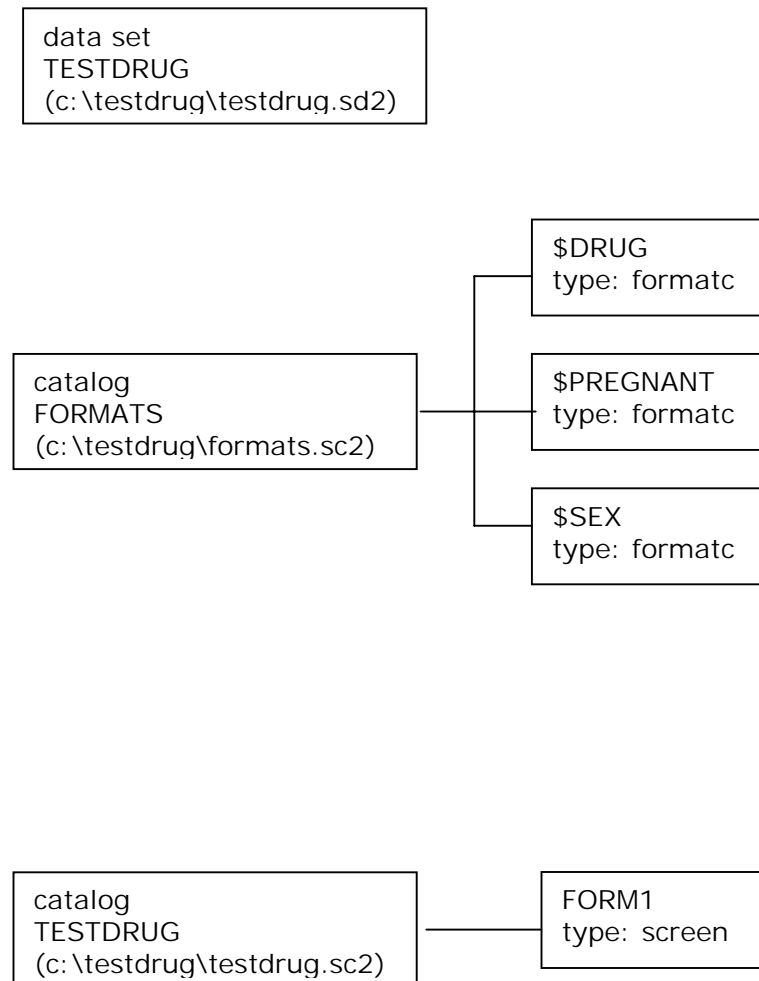
H:\fspclass\testdrug

## Summary

At this point these are the SAS members and entries we have created in the SAS library c:\testdrug:

### Members

### Entries



## Workshop 1

Using FSEDIT, create a SAS data set called DEMOG and a SAS catalog called DEMOG. In the catalog create a screen called FORM to enter the data from the forms shown on the following pages. Please save all the files in the directory: c:\sasfsp.

Modify the application as indicated in the guidelines below.

1. The patient ID must be entered.
2. Make the visit date a SAS date value.
3. Create an SCL program to display the value of the body mass index. You can calculate it using this formula:  $bmi = 703 * weight / height^2$ , where weight is in pounds and height is in inches. The value displayed should be rounded (hint: specify a format when you define it).
4. In the SCL program verify that:
  - a. the value of sex is correct (F or M)
  - b. the value of race is correct (1 through 5)
  - c. if the user enters a 5 (i.e. other) for the race then they should write the race in the field provided

The user should not be able to leave the observation if any of these errors are found.

## DEMOGRAPHICS

Patient ID 11

Visit Date 5/1/98 (mmddyy)

First Name Francisco

Last Name Toledo

Sex M

Race 3

F Female

M Male

1 Caucasian

2 Black

3 Hispanic

4 Asian

5 Other \_\_\_\_\_  
(specify)

Height (in) 68

Weight (pounds) 184

---

## DEMOGRAPHICS

Patient ID 13

Visit Date 4/29/98 (mmddyy)

First Name Rachel

Last Name Li

Sex F

Race 4

F Female

M Male

1 Caucasian

2 Black

3 Hispanic

4 Asian

5 Other \_\_\_\_\_  
(specify)

Height (in) 61

Weight (pounds) 105

## DEMOGRAPHICS

Patient ID 12

Visit Date 5/12/98 (mmddyy)

First Name Christopher

Last Name Rohde

Sex M

F Female

M Male

Race 1

1 Caucasian

2 Black

3 Hispanic

4 Asian

5 Other \_\_\_\_\_  
(specify)

Height (in) 69

Weight (pounds) 165

---

## DEMOGRAPHICS

Patient ID 14

Visit Date 4/30/98 (mmddyy)

First Name Elias

Last Name Bedewi

Sex M

F Female

M Male

Race 5

1 Caucasian

2 Black

3 Hispanic

4 Asian

5 Other lebanese  
(specify)

Height (in) 70

Weight (pounds) 176

## Setting the Function Keys

One way to set the function keys is to open the default KEYS window from the Help menu and modify that window then save it elsewhere. You can also use the BUILD window of SAS/AF. We will do it the first way.

Open the KEYS window. (= > KEYS) Modify it then select Save As from the File menu. In the blank field enter a three-part name for the function keys in the form: *libref.catalog.keyname*

(= > SAVE *libref.catalog.keyname*)

The libref and catalog name must be the same as the catalog where you saved the screen.

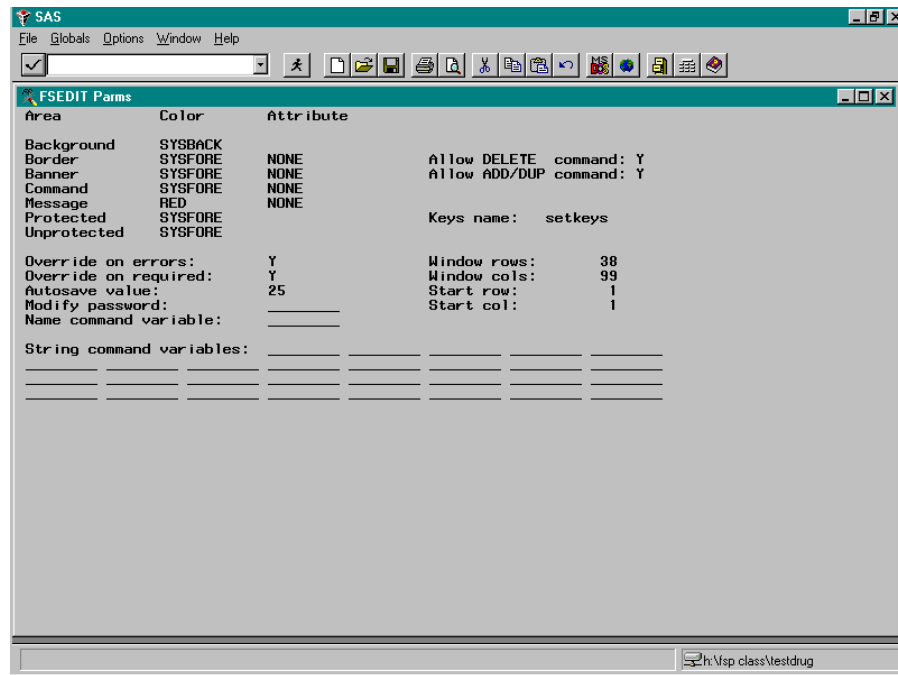
We will enter LIBRARY.TESTDRUG.SETKEYS then press OK. This creates the entry: SETKEYS.KEYS in the catalog TESTDRUG.



| Key     | Definition |
|---------|------------|
| F1      | help       |
| F2      | keys       |
| F3      | save       |
| F4      | cancel     |
| F5      | add        |
| F6      | dup        |
| F7      |            |
| F8      | end        |
| F9      |            |
| F11     |            |
| F12     |            |
| SHF F1  |            |
| SHF F2  |            |
| SHF F3  |            |
| SHF F6  |            |
| SHF F7  |            |
| SHF F8  |            |
| SHF F9  |            |
| SHF F10 |            |
| SHF F11 |            |

Now select Cancel from the File menu to close the KEYS window. (= > CANCEL) It is very important that you choose Cancel, otherwise you will override the default KEYS window permanently.

To associate these function keys with the screen select Modify screen from the Locals menu while you are in FSEDIT. (= > MODIFY) Then select item number 5 "Modification of General Parameters" from the FSEDIT Menu. (= > 5) In the field labeled "Keys name" enter the keys name: SETKEYS. You don't need the libref and catalog name. Then select End from the File menu. (= > END)



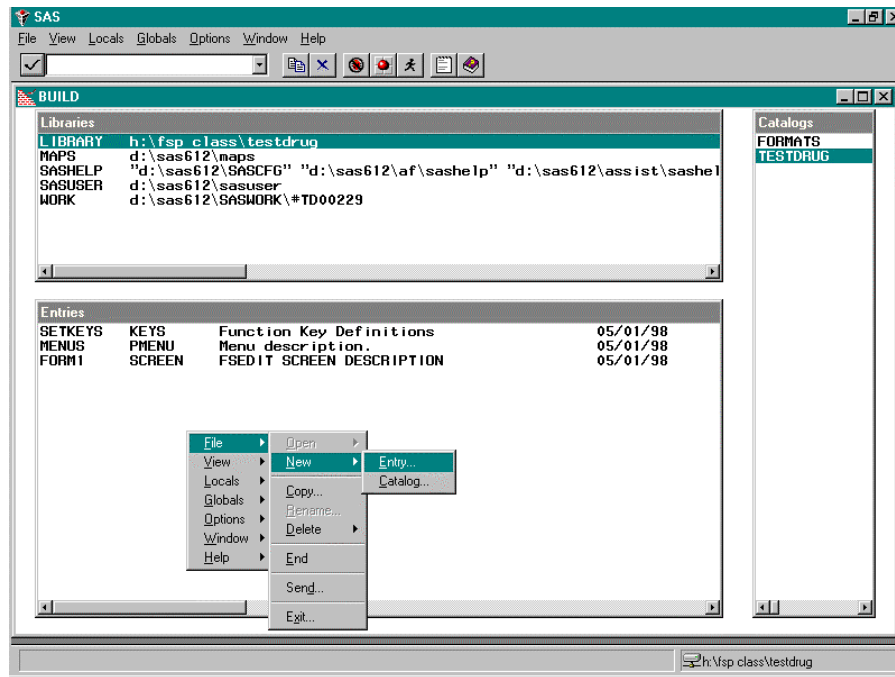
Since we would like to customize our application with a help window and pull-down menus we will exit FSEDIT and enter the data later. Press the GOBACK button (= > END) then select End from the File menu to return to the Program Editor (= > END).

## Creating a Help Window

You now have the option of creating a help window using the BUILD window that is part of the SAS/AF product.

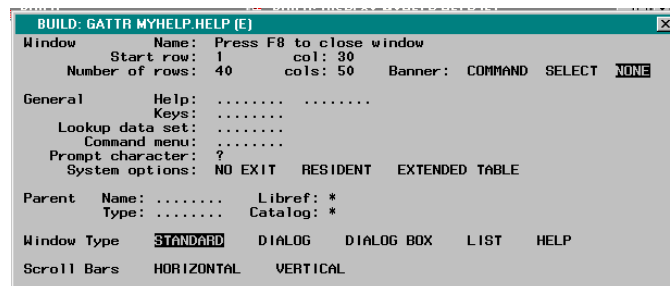
To open the BUILD window select the Globals menu then Develop then Application builder. (= > BUILD)

From the Libraries list select LIBRARY. From the Catalogs list select TESTDRUG. Now press the right mouse button and select File then New then Entry. (= > EDIT library.testdrug.myhelp.help)



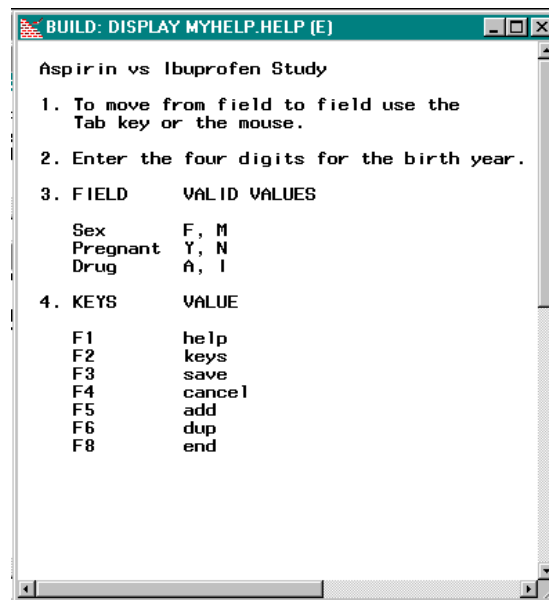
For the Entry name we type MYHELP then we select HELP from Entry type drop-down list then press the OK button. An empty BUILD window titled MYHELP.HELP will be displayed. Here you can enter any information that will help the user use the application. If you have saved help information in a text file you can include the file by selecting Open from the File menu then Read file. (= > INC 'filename')

To customize the help window select Set general attributes from the Locals menu and make the necessary changes. (= > GATTR) Select End from the File menu when done. (= > END)





Here is what our help window looks like:



When you are done, select End from the File menu to close the Help window. (= > END) Now the Entries list shows the new entry MYHELP.HELP. Select End from the File menu to exit the BUILD window and return to the Program Editor window. (= > END)

We associate this help entry with the FSEDIT application by adding the statement shown below to the FSEINIT section of the SCL program:

```
call execcmdi('sethelp library.testdrug.myhelp.help');
```

## Creating Pull-Down Menus

Base SAS provides a procedure called PMENU that you can use to build pull-down menus. You can later associate these menus with your FSEDIT application. You can find the full documentation of PROC PMENU in the "SAS Procedures Guide" or through the Help facility in SAS.

```
proc pmenu c=library.testdrug;  
menu menus; ❶  
  item 'File'    menu=f mnemonic='F'; ❷  
  item 'Edit'    menu=e mnemonic='E';  
  item 'Search' menu=s mnemonic='S';  
  item 'Go to'   menu=g mnemonic='G';  
  item 'Help'    menu=h mnemonic='H';
```

```

menu f; ❸
    item 'Save'          mnemonic='S'; ❹
    item 'Cancel'        mnemonic='C';
    item 'Print Screen'  mnemonic='P' selection=p;
    item 'End'           mnemonic='E' dialog=e;
        selection p 'sprint'; ❺
        dialog e 'End'; ❻
        text #2 @5 'Are you sure you want to exit?';

menu e;
    item 'Add'           mnemonic='A';
    item 'Duplicate'     mnemonic='P' selection=dup;
    item 'Delete'        mnemonic='D' dialog=d;
        selection dup 'dup';
        dialog d '%1'; ❼
            text #2 @1 'Delete Record?';
            radiobox default=1;
                rbutton #3 @1 'Cancel';
                rbutton #4 @1 'Delete';

menu s;
    item 'Subject name sounds like' mnemonic='S' dialog=w1;
    item 'Subject number'          mnemonic='N' dialog=w2;
    separator; ❽
    item 'Undo search'              mnemonic='U' selection=u;
        dialog w1 'where NAME ? upcase("@1)";
            text #2 @8 len=30;
        dialog w2 'where SUBNO = upcase("@1)"; ❾
            text #2 @1 'Enter subject number:';
            text #3 @8 len=4;
        selection u 'where undo';

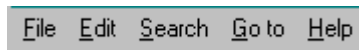
menu g;
    item 'First'           mnemonic='F' selection=f;
    item 'Last'            mnemonic='B' selection=l;
    item 'Next'            mnemonic='N' selection=n;
    item 'Previous'        mnemonic='P' selection=p;
        selection f 'top';
        selection l 'bottom';
        selection n 'forward';
        selection p 'backward';

menu h;
    item 'How to enter data' mnemonic='H' selection=h;
        selection h 'help';

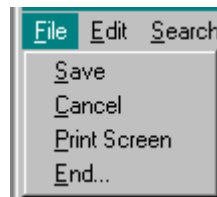
run;
quit;

```

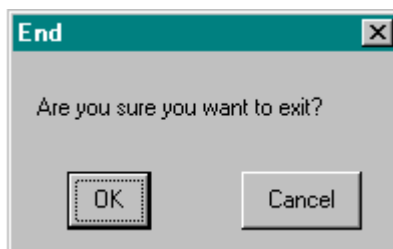
- ❶ This MENU statement is used to create a catalog entry called MENU. This entry will contain our pull-down menus.
- ❷ These ITEM statements define the menu names that will appear at the top of the application. The option MENU= specifies the label of the secondary MENU statements that define the entries within each pull-down menu. The MNEMONIC= option defines a single character that can be typed instead of clicking on the menu item.



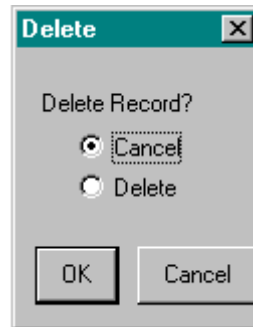
- ❸ The secondary MENU statements define the entries of each pull-down menu.



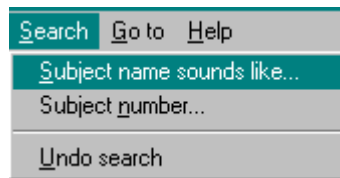
- ❹ These ITEM statements define the entries of a pull-down menu and the action taken when an entry is selected. You can use the SELECTION= or DIALOG= options to link to other statements that define an action. By default, if you specify a valid Display Manager command in quotes and do not use the SELECTION= or DIALOG= options then SAS will execute that command.
- ❺ The SELECTION statement includes a command that is executed when the user chooses that selection.
- ❻ The DIALOG statement defines a dialog box. The dialog box includes text and may either contain a radiobox or a checkbox. Use a radiobox for mutually exclusive choices and a checkbox for multiple choices. The DIALOG statement must be followed by at least one TEXT statement that includes a string that will appear in the dialog window. Then it can be followed by RADIOBOX, RBUTTON or CHECKBOX statements. SAS includes an OK button and a Cancel button. Here, the question "Are you sure you want to exit?" will be displayed. The user can press either the OK or the Cancel button.



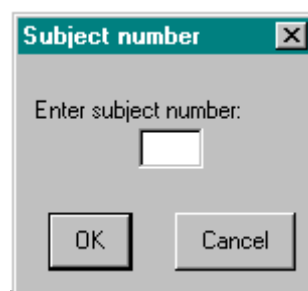
- ⑦ This DIALOG statement displays a dialog box with a radiobox. The question "Delete record?" is displayed with two options underneath: Cancel or Delete. A RADIOBOX statement must always define a default selection, in this case it is Cancel which is selection number 1. The string "%1" contains the value of the option the user selects. This notation is used only with RADIOBOX statements. The RBUTTON statements define the different choices.



- ⑧ The SEPARATOR statement draws a line between items in a menu.



- ⑨ In this dialog box the text "Enter subject number:" appears with a blank field underneath. The field is created by using the LEN= option in the TEXT statement. The string "@1" contains the value that the user enters. This notation is only used with the TEXT statement.



To associate these menus with the FSEDIT application we add the statement shown below to the FSEINIT section of the SCL program:

```
call execcmdi('setpmenu library.testdrug.menus.pmenu');
```

## Modified SCL Program

Here is the modified FSEINIT section of the SCL program. The rest of the program remains the same.

```
FSEINIT:
  control error label;
  call execcmdi('setpmenu library.testdrug.menus.pmenu');❶
  call execcmdi('sethelp library.testdrug.myhelp.help');❷
  call execcmdi('pgm off; log off; listing off');❸
  call wname(' ');❹
return;
```

- ❶ Specifies the pull-down menus to use.
- ❷ Specifies the help window to use.
- ❸ Eliminates the PGM, LOG and OUTPUT windows.
- ❹ Removes the default FSEDIT title (i.e. name of SAS data set).

## Summary

These are all the SAS members and entries of our FSEDIT application:

### Members

data set  
TESTDRUG  
(c:\testdrug\testdrug.sd2)

catalog  
FORMATS  
(c:\testdrug\formats.sc2)

catalog  
TESTDRUG  
(c:\testdrug\testdrug.sc2)

### Entries

\$DRUG  
type: formatc

\$PREGNANT  
type: formatc

\$SEX  
type: formatc

FORM1  
type: screen

SETKEYS  
type: keys

MYHELP  
type: help

MENUS  
type: menu

## Setting Up an Icon for the Application

We would like to create an icon that the end user can double-click to enter data. To make it very easy for the user we will eliminate any references to the SAS System. Double-clicking on the icon would take them directly to the data entry form then when they exit from the application it would place them back in Windows.

First create a program called TESTDRUG.SAS as shown below that invokes the application. Notice that the PROC statement now says DATA= and not NEW= since the data set already exists.

```
dm 'command close; toolclose; wstatusln off';❶  
libname library 'c:\testdrug';  
proc fsedit data=library.testdrug screen=library.testdrug.form1;  
run;  
endsas;❷
```

- ❶ The DM statement executes the three Display Manager commands shown to eliminate the command bar, the toolbar and the status line.
- ❷ The ENDSAS statement ends the SAS session immediately after the FSEDIT session is over.

Next, copy the CONFIG.SAS file that is located in your SAS root directory to the application directory, i.e. TESTDRUG. Modify the lines that specify -sasuser and -work so they point to the application's directory. Then add the other lines shown here to customize the appearance of the screen. You can add these lines to the top of the program:

```
-sasuser c:\testdrug\sasuser ❶  
-work c:\testdrug\saswork use ❷  
-nosplash ❸  
-awsdef 10 10 79 75 ❹  
-awstitle 'Drug Study: Aspirin vs Ibuprofen' ❺  
-noawsmenumbermerge ❻  
-awscontrol nosystemmenu nominmax ❼  
-sascontrol nosystemmenu nominmax ❽
```

- ❶ Defines the directory where the SASUSER profile is located.
- ❷ Defines the work directory.
- ❸ Eliminates the SAS logo that appears at invocation.

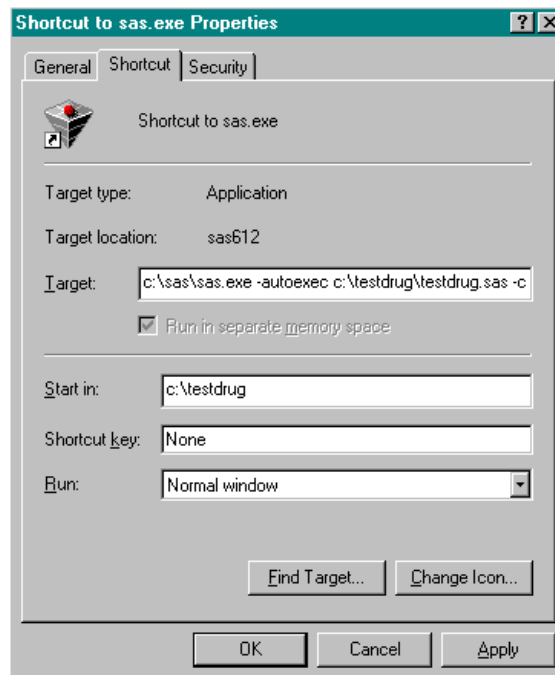
- ④ Defines the size of the SAS window in screen percentages: start\_row\_pct, start\_col\_pct, height\_pct, width\_pct.
- ⑤ Gives the SAS window a title.
- ⑥ Eliminates the default SAS menus.
- ⑦ Eliminates system menu and min/max buttons from the SAS Display Manager.
- ⑧ Eliminates system menu and min/max buttons from the application window (i.e. FSEDIT).

You can create an icon for the FSEDIT application by creating a shortcut of the SAS.EXE file. Locate your SAS directory and drag SAS.EXE to the desktop. Single-click on the icon then press the right-mouse button and select Properties. In the Target field enter:

```
c:\sas\sas.exe -autoexec c:\testdrug\testdrug.sas -config c:\testdrug\config.sas
```

This assumes you installed SAS in the C:\SAS directory. If you didn't then modify the line so it points to the correct drive and directory.

In the Start field enter: c:\testdrug, then press the Change icon button if you want to select a different icon.



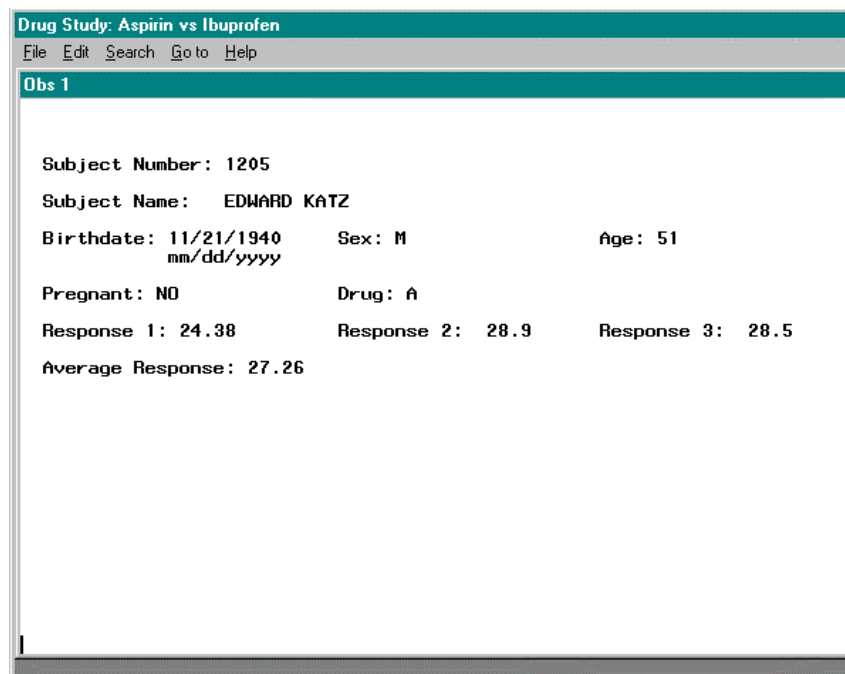


Now your users can just double-click on the icon you created to start entering or browsing the data.



When the user double-clicks on the icon the FSEDIT application will open automatically showing the customized screen and menus. When they end the application, FSEDIT and the SAS session will both be terminated, leaving the user back in Windows.

This is what the application looks like when the user opens it:

A screenshot of a Windows application window titled "Drug Study: Aspirin vs Ibuprofen". The window has a menu bar with "File", "Edit", "Search", "Go to", and "Help". Below the menu bar is a teal header bar with "Obs 1". The main area of the window displays the following data:

|                           |                  |                  |
|---------------------------|------------------|------------------|
| Subject Number: 1205      |                  |                  |
| Subject Name: EDWARD KATZ |                  |                  |
| Birthdate: 11/21/1940     | Sex: M           | Age: 51          |
| mm/dd/yyyy                |                  |                  |
| Pregnant: NO              | Drug: A          |                  |
| Response 1: 24.38         | Response 2: 28.9 | Response 3: 28.5 |
| Average Response: 27.26   |                  |                  |

They can then use the menus to browse, edit or add data.

## Workshop 2

1. Create a Help window. Save it in the catalog DEMOG.
2. Create pull-down menus by running the program `c:\sasfsp\expmenus.sas`. Make sure you modify the statements indicated in the enclosed comment.
3. Now go back to FSEDIT and modify the SCL program so it includes the Help window and the pull-down menus you created.

## Optional

4. Create an icon for the application. You will need to modify the CONFIG.SAS file. Please use the CONFIG.SAS file located in the directory `c:\sasfsp`. Do not use the one in the `c:\` directory!

## Appendix: Useful FSEDIT Commands

| Key    | Command     | Description                  |
|--------|-------------|------------------------------|
| F1     | help        | displays FSEDIT help         |
| F3     | end         | closes window                |
| F5     | add         | adds a new observation       |
| F6     | dup         | duplicates observation       |
| F7     | modify      | opens the modify window      |
| F8     | end         | same as F3                   |
| F9     | keys        | opens the FSEDIT keys window |
| F11    | command bar | go to the command bar        |
| Shf F7 | left        | scroll to the left           |
| Shf F8 | right       | scroll to the right          |

## Solutions to Workshops

### Workshop 1

FSEDIT program to create SAS data set and screen:

```
libname library 'c:\sasfsp';  
proc fsedit new=library.demog screen=library.demog.form label;  
run;
```

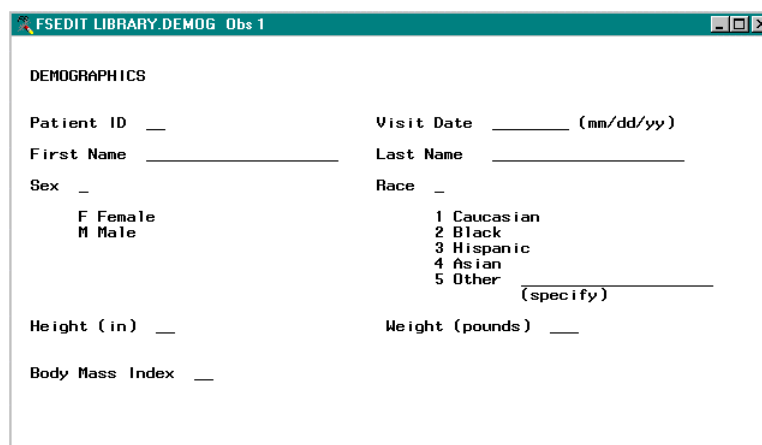
You can define the variables attributes in FSEDIT's New window as follows:

| Name     | Type | Length | Label      | Informat | Format   |
|----------|------|--------|------------|----------|----------|
| id       | c    | 2      | Patient ID |          |          |
| visitdt  | n    |        | Visit Date | mmddyy8. | mmddyy8. |
| fname    | c    | 20     | First Name |          |          |
| lname    | c    | 20     | Last Name  |          |          |
| sex      | c    | 1      |            |          |          |
| race     | c    | 1      |            |          |          |
| race_oth | c    | 20     |            |          |          |
| height   | n    |        |            |          |          |
| weight   | n    |        |            |          |          |

The attributes of the SCL variable BMI (i.e. body mass index) can be defined as:

| Name | Type | Length | Label | Informat | Format |
|------|------|--------|-------|----------|--------|
| bmi  | n    |        |       |          | 2.     |

The screen could look like this:



The screenshot shows a window titled "FSEDIT LIBRARY.DEMOG Obs 1". Inside, under the heading "DEMOGRAPHICS", there are several input fields and lists. The fields are arranged in two columns. The first column contains "Patient ID", "First Name", "Sex", "Height (in)", and "Body Mass Index". The second column contains "Visit Date (mm/dd/yy)", "Last Name", "Race", and "Weight (pounds)". The "Sex" field has a list with "F Female" and "M Male". The "Race" field has a list with "1 Caucasian", "2 Black", "3 Hispanic", "4 Asian", and "5 Other", followed by a "(specify)" label and an input line. The "Visit Date" field has a format label "(mm/dd/yy)".

SCL program:

```
FSEINIT:
  control label;
return;

INIT:
  bmi=703*weight/height**2;
return;

SEX:
  if sex not in('F','M') then do;
    erroron sex;
    _msg_='Sex must be F or M, please reenter.';
  end;
return;

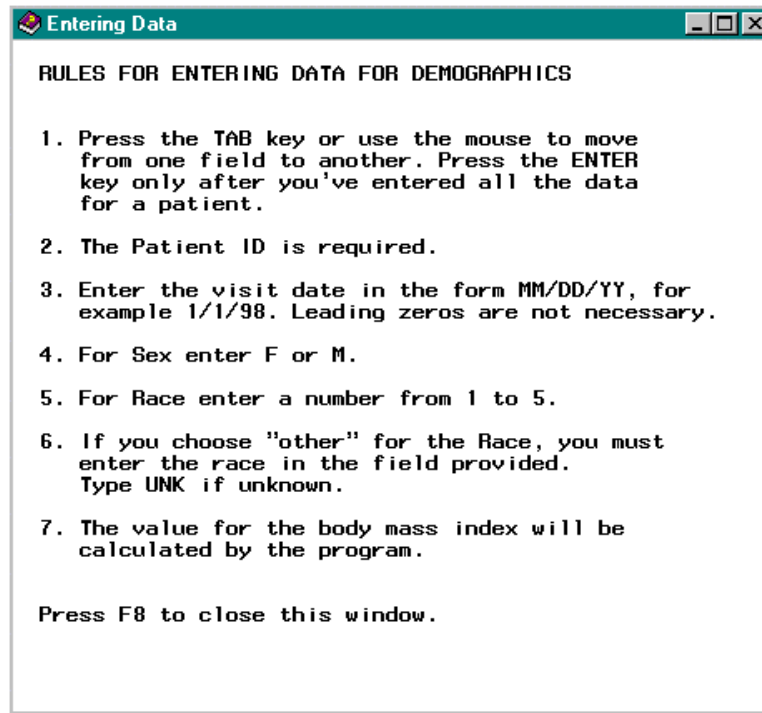
RACE:
  if race not in('1','2','3','4','5') then do;
    erroron race;
    _msg_='Race must be a number from 1 to 5, please reenter.';
  end;
return;

MAIN:
  if race='5' and race_oth=' ' then do;
    erroron race_oth;
    _msg_='You must specify a race. Type UNK if unknown.';
    return;
  end;
  bmi=703*weight/height**2;
return;

TERM:
return;
```

## Workshop 2

1. From the Globals menu select Develop then Application builder. Select the libref and the catalog where your application is saved. Press the right-mouse button and select File then New then Entry. Give the entry a name (e.g. MYHELP), choose HELP for the type and click OK. Enter the information you want to include in the Help window, for example:



Select End from the File menu. Again, select End from the File menu.

2. This is the PMENU program. Replace LIBREF by the libref you used in the LIBNAME statement.

```
proc pmenu c=LIBREF.demog;  
menu menus;  
  item 'File'      menu=f mnemonic='F';  
  item 'Edit'      menu=e mnemonic='E';  
  item 'Search'    menu=s mnemonic='S';  
  item 'Go to'     menu=g mnemonic='G';  
  item 'Help'      menu=h mnemonic='H';  
  
  menu f;  
    item 'Save'          mnemonic='S';  
    item 'Cancel'        mnemonic='C';  
    item 'Print Screen'  mnemonic='P' selection=p;  
    item 'End'           mnemonic='E' dialog=e;
```

```

        selection p 'sprint';
        dialog e 'end';
        text #2 @5 'Are you sure you want to exit?';

menu e;
    item 'Add'          mnemonic='A';
    item 'Duplicate' mnemonic='P' selection=dup;
    item 'Delete'       mnemonic='D' dialog=d;
        selection dup 'dup';
        dialog d '%1';
            text #2 @1 'Delete Record?';
            radiobox default=1;
                rbutton #3 @1 'Cancel';
                rbutton #4 @1 'Delete';

menu s;
    item 'Last name sounds like' mnemonic='L' dialog=w1;
    item 'Patient ID'           mnemonic='I' dialog=w2;
    separator;
    item 'Undo search'          mnemonic='U' selection=u;
        dialog w1 'where LNAME ? upcase("@1")';
            text #2 @8 len=30;
        dialog w2 'where ID = upcase("@1")';
            text #2 @1 'Enter Patient ID: ';
            text #3 @8 len=4;
        selection u 'where undo';

menu g;
    item 'First'          mnemonic='F' selection=f;
    item 'Last'           mnemonic='L' selection=l;
    item 'Next'           mnemonic='N' selection=n;
    item 'Previous'       mnemonic='P' selection=p;
        selection f 'top';
        selection l 'bottom';
        selection n 'forward';
        selection p 'backward';

menu h;
    item 'How to enter data' mnemonic='H' selection=h;
    selection h 'help';

run;
quit;

```

3. Here's the modified FSEINIT section. Replace LIBREF by the libref you used in the LIBNAME statement.

```
FSEINIT:
  control label;
  call execcmdi('setpmenu LIBREF.demog.menus.pmenu');
  call execcmdi('sethelp  LIBREF.demog.myhelp.help');
return;
```

4. Follow the example in the section "Setting Up an Icon for the Application" making the necessary changes so it uses the names you used in your application.